Summary of Nek5000 Simulations on BGW

Nek5000 is a spectral-element-based code for simulating hydrodynamics, magnetohydrodynamics, and heat transfer that is being used by dozens of groups worldwide for a broad range of problems including aeroacoustics, astrophysics, blood flow modeling, combustion, geophysics, multiphase flow, reactor cooling, and turbulence applications. Nek5000 is based on a semi-implicit formulation of the incompressible Navier-Stokes equations and requires the solution of a set of elliptic PDEs at each timestep. The system for the pressure, which enforces incompressiblity throughout the domain, is solved using a state-of-the art multigrid algorithm that features a scalable fast coarse-grid solver [Tufo and Fischer, 2001]. The code was developed as one of the first available applications for distributed memory parallel processors and has demonstrated scalability to 4096 processors on ASCI Red (which was recognized with a 1999 Gordon Bell Prize) and to 2048 processors on BGL.

In the spectral element method, one breaks the domain into E computational elements and approximates the solution within each element by tensor-product polynomial bases of degree N. The use of high-order (typ. N=8–16) provides rapid numerical convergence, while tensor-product forms lead to processor efficiencies deriving from the use of locally lexicographical (ijk) indexing and the use of matrix-matrix products to efficiently apply differential operators. Parallelism in the spectral element method is realized through a standard domain decomposition approach in which the unstructured collection of elements is distributed in contiguous subgroups to individual processors.

For BGW Day, we simulated an astrophysical MHD problem, known as the magneto-rotational instability (MRI), that relates to momentum transport in accretion disks. The setup consists of a pair of concentric cylinders that are rotating in such a way that the flow is nominally hydrodynamically stable, but becomes unstable in the presence of a magnetic field. This project is being supported by the NSF Physics Frontier Center for Magnetic Self Organization and by a 2005 DOE INCITE award, for which we have 2 million node hours at NERSC. Substantial access to BGW will allow us to go to higher Reynolds numbers than being considered under the current INCITE project. Complementary experiments are being conducted by CMSO team members Ji and Goodman at the Princeton Plasma Physics Laboratory.

Goals

We have two primary goals. One is to demonstrate/investigate the scalability of our algorithms. The other is to investigate the MRI at a Reynolds number of Re=62,000, which is a factor of ten higher than our current INCITE investigation. What is important is that we will be able to investigate much higher magnetic Reynolds numbers ($Rm \approx$ 31,000 in the proposed case) than will ever be possible in any terrestial experiment.

Results

On BGW day, we were able to make several runs on up to 8192 processors, in both virtual node (VN) and co-processor (CO) modes. Because of limited time, we focussed on one particular mesh comprising E=15360 elements for polynomial degrees N=7 and 9. (Larger

P	n_v	n_p	A	A'	B'	C'	D'	n_{step}	t_{run}
2048u	20.3M	15.4M	430.7	1.44	24.99	165.9	282.6	100	838.4
4096c	20.3M	15.4M	941.3	3.31	54.87	362.0	617.5	100	467.9
8192c	20.3M	15.4M	2088.	7.97	122.0	805.9	1374.	100	282.1
8192v	20.3M	15.4M	3496.	12.2	20.22	1349.9	2303.	100	341.5
1024u	5.2M	3.3M	74.6	.64	11.3	35.9	74.5		
2048u	5.2M	3.3M	165.8	1.45	25.0	79.6	165.7		
4096u	5.2M	3.3M	362.9	3.31	54.8	173.9	362.4	100	127.3
8192v	5.2M	3.3M	1337.	12.4	203.2	647.6	1345	100	114.4

meshes will be used for runs on $P > 8192$.) The timing results are presented in Table 1. The number of processors are given in the first column, with a suffix c for co-processor mode, v for VN mode, and u for VN mode in which the block is undersubscribed (i.e., the second processor on each node is idle). Entries with the c and u suffix are thus comparable. The number of velocity and pressure points are indicated in the second and third columns, respectively. The simulations that completed ran for 100 timesteps and the runtime, over the course of the simulation, is given in the last column. The entries in columns A and A'–D' will be discussed below. We were able to complete one strong scaling study for P=2048, 4096, and 8192 with 20.3 M grid points. Assuming unity efficiency for the smallest case, we have an upper bound on parallel efficiency of $\eta(P = 4096) = (838.4/467.9)/2 = .90$ and $\eta(P = 8192) = (838.4/282.1)/4 = .75$. The true efficiency will be somewhat less, but not much.

The efficiency of $\eta < .75$ for the $P = 8192$ is very much in line with what can be expected for a computation of this type when there are only 2500 points per processor. For the ANL-hosted Petascale Workshop held last March, we developed a computational model that predicts the performance of several algorithms for solving Poisson's equation on distributed memory platforms. (The model is designed only to reflect parallel efficiency; algorithmic efficiency is a separate concern.) The constants in the model were based on timing measurements made on BGL at Argonne. Table 2 shows the predictions for four algorithms at a point-per-processor loading of n_v/P=2478. The algorithms have increasing communication complexity; point Jacobi iteration requires only nearest-neighbor exchanges; conjugate gradient (CG) requires vector reductions; multigrid (MG) requires global coarse grid solves. Two flavors of MG are modeled. One (STD) involves the standard coarse-grid solve approach that requires $O(n_c)$ communication per iteration, where n_c is the dimension of the coarse grid system and one (XX^T) requires only $O(n_c^{2/3})$ communication. The latter approach was used for the pressure solve in our computations. In the present computations, roughly 48% of the time was spent in the pressure solve (similar to MG-XX^T) and 43% of the time was spent in the velocity solve (similar to CG). Our computations have somewhat more work per iteration than the model, which is based on a simple 7-point finite difference method. Finally, the model efficiencies for CG and MG are probably a bit low because the vector reduction (allreduce) estimates were based on communication times measured on the torus rather than the tree.

We return to the columns A, A'–D' of Table 1. These represent startup costs, in seconds, for each of the runs considered. Column A is for the primary data exchange and columns A'–

Table 2: BGL Efficiency Estimates for Poisson Problem

Method	$\eta(P)$	n_v/P	P
Jacobi	0.9057	2478	8192
CG	0.7009	2478	8192
MG-XX^T	0.4399	2478	8192
MG-STD	0.1212	2478	8192

Table 3: Gather-Scatter Code Setup Time (s)

P	t_{old}	t_{new}	n_v/P	n_v
64c	2.7366E+00	1.19326E+00	122880	7864320
128c	5.7935E+00	6.66121E-01	61440	7864320
256c	1.2614E+01	3.70891E-01	30720	7864320
512c	2.9149E+01	2.25903E-01	15360	7864320
128u	5.5554E+00	6.70362E-01	61440	7864320
256u	1.2115E+01	3.72556E-01	30720	7864320
512u	2.8182E+01	2.26880E-01	15360	7864320
1024u	6.3762E+01	1.79818E-01	7680	7864320
2048v	2.2938E+02	3.02135E-01	3584	7864320
128c	9.8481E+00	1.20675E+00	120000	15360000
256c	2.1855E+01	6.68387E-01	60000	15360000
512c	5.0286E+01	3.94396E-01	30000	15360000
256u	2.0695E+01	6.70618E-01	60000	15360000
512u	4.8030E+01	3.96010E-01	30000	15360000
1024u	1.0885E+02	2.79369E-01	15000	15360000
2048v	3.8777E+02	3.59496E-01	7000	15360000

D' represent setup times for the multigrid sublevels. Our nearest-neighbor (gather-scatter) exhanges (which are nearest-neighbor only in the computational mesh and not on the BGL torus) require point-to-point and processor-to-processor maps that can only be generated at run time. By design, the interface to the execution and set up phases of this central communication kernel is very simple and the code has found significant use in applications outside Nek5000. The setup time with the current implementation, however, is linear in P and linear in n_v. The constant is sufficiently small that this scaling is not a concern for relatively small problems on up to $P = 2048$. Typical run times for our simulations are on the order of 24 hours, so even a few minutes of setup time is not a particular concern for the number of processors that we typically have access to. It did, however, significantly impact our ability to make scaling tests on BGW.

The orginal author of the gather-scatter (gs) code was unavailable when we were notified of access to BGW, so we started to rewrite the gs code from scratch, in anticipation of this known scaling limitation. Unfortunately, we were not able to complete this in time for BGW day. We have since, however, been able to time the new setup phase on Argonne's BGL at up to $P = 2048v$. Setup times, in seconds, for this new code are given in Table 3. It can be seen that the new code, which is based on the complete exchange strategy described in [Deville, Fischer, and Mund, 2002], realizes times that initially scale as $1/P$, rather than P.